AFRL-RI-RS-TR-2015-051

MODELING & ANALYSIS OF MULTICORE ARCHITECTURES FOR EMBEDDED SIGINT APPLICATIONS

MARCH 2015

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the 88th ABW, Wright-Patterson AFB Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2015-051 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/ S /
JOSEPH A. CAROLI
Chief, High Performance Systems Branch

/ S /
MARK H. LINDERMAN
Technical Advisor, Computing & Communications Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
MARCH 2015	FINAL TECHNICAL REPORT	APR 2013 – SEP 2014

4. TITLE AND SUBTITLE

MODELING & ANALYSIS OF MULTICORE ARCHITECTURES FOR EMBEDDED SIGINT APPLICATIONS

5a. CONTRACT NUMBER
IN-HOUSE – R101

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Ryan S. Luley

5d. PROJECT NUMBER
T2MM

5e. TASK NUMBER
IN

5f. WORK UNIT NUMBER
HO

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/Information Directorate
Rome Research Site/RITB
525 Brooks Road
Rome NY 13441-4505

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/Information Directorate
Rome Research Site/RITB
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2015-051

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. PA# 88ABW-2015-1074
Date Cleared: 16 MAR 2015

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The future battlespace is likely to be increasingly contested and, in many cases, completely denied to joint military forces. Traditional operational approaches, including methods for intelligence, surveillance, and reconnaissance (ISR), will be challenged by the shift from permissive to non-permissive domains. The utility of embedded processing architectures will be driven by energy efficiency as much as it will be by high performance. Fortunately, there has been tremendous growth in the development of high performance, low power multi- and many-core architectures. This project sought to develop a power and performance modeling approach to apply towards such emerging architectures. Through this modeling approach, the intent is to (1) accurately predict peak application performance, as opposed to relying only on theoretical analysis and (2) identify optimal processor requirements, so as to minimize power consumption and/or more efficiently task processing resources. The capability offered by this modeling technique is expected to allow system designers to make more informed selection of high performance embedded computing (HPEC) technologies.

15. SUBJECT TERMS
analytical performance model, embedded computing, energy efficient computing, high performance computing, multicore computing, power modeling, signals intelligence, signal processing, Wigner-Ville distribution

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**RYAN S. LULEY**
U	U	U	UL	28	19b. TELEPHONE NUMBER (*Include area code*) **N/A**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

It would not have been possible to complete this project without the assistance of several colleagues. I would like to thank David Gomez for his assistance with executing and analyzing kernel performance on the Parallella processor, first as a student intern and then as a contractor with Rome Research Corporation. I would also like to thank the AFRL/RITB High Performance Computing Help staff – Alex Stuart, Cameron Baker, and Chris Urich, all of Rome Research Corporation – for assisting with my regular system administration requests and ensuring that I always had the tools necessary to complete this research. I would like to thank Clare Thiem for generously loaning multiple Parallella boards from his in-house project for evaluation and testing under this project. Last, but certainly not least, I would like to thank Dr. David Richie of Brown Deer Technology for his guidance in utilizing COPRTHR and for his development efforts to port the SRAD algorithm to STDCL. Dr. Richie's support was made possible through the Department of Defense High Performance Computing Modernization Program's User Productivity Enhancement, Technology Transfer and Training (PETTT) initiative.

1. SUMMARY

The future battlespace is likely to be increasingly contested and, in many cases, completely denied to joint military forces. Traditional operational approaches, including methods for intelligence, surveillance, and reconnaissance (ISR), will be challenged by the shift from permissive to non-permissive domains. It is believed that a new set of ISR capabilities – referred to as non-traditional ISR (NTISR) – will be needed. Many of these NTISR techniques lack the sensor capability and/or are constrained by size, weight, and power (SWAP) limitations, which will force the USAF to consider new approaches to processing, exploitation, and dissemination (PED). As one example, the utility of embedded processing architectures will be driven by energy efficiency as much as it will be by high performance. Fortunately, there has been tremendous growth in the development of high performance, low power multi- and many-core architectures. While recent PED research space has been largely dominated by general purpose graphics processing units (GPGPUs), there is evidence that the GPGPU is not a "silver bullet" and other architectures must be considered.

This project sought to develop a power and performance modeling approach to apply towards such emerging architectures. Through this modeling approach, the intent is to (1) accurately predict peak application performance, as opposed to relying only on theoretical analysis and (2) identify optimal processor requirements, so as to minimize power consumption and/or more efficiently task processing resources. The capability offered by this modeling technique is expected to allow system designers to make more informed selection of high performance embedded computing (HPEC) technologies. Furthermore, it could allow researchers to design resource management and PED techniques for managing whole-system optimizations in networks of heterogeneous HPEC architectures.

2. INTRODUCTION

Recent PED research and development for military ISR has been driven by the idea that the amount of data being collected is far outpacing the ability to process it into actionable information in an expedient manner. Lt. Gen David Deptula has been credited with coining the phrase "swimming in sensors, drowning in data" that has been used to motivate the need for massive data analytics and extreme-scale computing technologies [1]. However a very

significant shift is on the horizon as it pertains to military ISR collection strategy. The operational environment is transforming into one that is much less permissive than seen in past conflicts. The USAF Scientific Advisory Board (SAB) defines two domains outside the permissive environment – contested and denied – that will need increased attention from the entire Planning and Direction, Collection, Processing and Exploitation, Analysis and Production, Dissemination (PCPAD) technology development community [2].

Joint Publication 3-0 defines a permissive environment as an "(o)perational environment in which host country military and law enforcement agencies have control as well as the intent and capability to assist operations that a unit intends to conduct" [3]. It can be assumed then that contested and denied, i.e. non-permissive, environments represent those in which a host country exhibits an increasing lack of control and/or cooperation. In these environments, there will be significant constraints on the typical collection and dissemination approaches used in permissive domains. Sensor products that exist in the permissive domain – e.g. full motion video (FMV), wide area motion imagery (WAMI), or radar – will be difficult to obtain in a contested environment. Instead, there is expected to be an increased reliance on NTISR platforms capable of collecting different classes of intelligence data, such as signals intelligence (SIGINT). Furthermore, communications networks will be very limited or worse, severely degraded. The ability to relay data to distributed ground processing stations for PED will be either very limited or impossible. Yet it will be of critical importance to assure the mission and thus the sensor network must be agile and resilient to such external factors. Therefore, it will be necessary to rethink the types of data that can be collected, as well as how that data is processed into information that can be immediately used to aid decision superiority over the battlefield.

SIGINT sensors are considered to be among the more likely assets that will be available within non-permissive domains of the future. Indeed, such approaches are well-suited to the types of ad hoc or opportune collection that fits the NTISR mold [4]. SIGINT involves "intelligence derived from electronic signals and systems used by foreign targets, such as communications systems, radars, and weapons systems… [and] provides a vital window for our nation into foreign adversaries' capabilities, actions, and intentions" [5]. Specifically, SIGINT techniques can be used to counter adversary systems that are designed for stealthy operation. Such adversary systems (e.g. low-probability-of-intercept (LPI) and low-probability-of-detection (LPD) radars) are intended to operate such that it is very difficult to determine location, intent, or

other characteristics that can be exploited in an effort to defeat the system. The ability to detect and locate LPI/LPD systems requires significant processing capability in order to enable real-time analysis and decision-making.

New HPEC technologies (Table 1) can offer the appropriate mix of performance and SWAP controls that would allow more efficient migration of PED techniques to NTISR platforms in the non-permissive domain. Processing must be pushed closer to the sensor in these cases in order to accomplish mission objectives efficiently and effectively, and an assessment of the tradeoffs between power and performance will be critical. Such assessments can be made by developing analytical models for the power and performance of emerging architectures. Empirical analysis will be too costly and slow, while theoretical analysis will likely result in overestimation of real-world capability that can lead to significant performance degradation. Furthermore, models that can be implemented in simulation would be beneficial because they would allow large-scale analysis of the effectiveness of techniques for resource management, sensor/processor deployment, and workload balancing within a heterogeneous network of HPEC systems.

Table 1 - Comparison of Selected HPEC Multicore Processors

Processor	**Cores**	**Speed (MHz)**	**Power (W)**	**SP Performance (GFLOPS)**	**Efficiency (GFLOPS/ W)**
Tilera TILEPro64 [6]	64[a]	700 – 866	19-23[b]	443[c]	19[d]
NVIDIA Tesla C2050/C2070 [6]	448[e]	1150	238	1030	4.3
NVIDIA Kepler K20 [7][8]	2496[e]	706	225	3520	15.6
Intel Xeon Phi 5110P [9]	60	1050	225	1010	4.5
Adapteva Epiphany [10]	16 – 4K	800	0.270	19	70.4
NVIDIA CUDA on ARM Architecture (CARMA) (Tegra 3 ARM A9 + NVIDIA Quadro 1000M) [11][12][13]	*Tegra:* 4 *Quadro:* 96[e]	*Tegra*: 1600 *Quadro*: 700	*Tegra*: 2 *Quadro*: 45	270	6

[a] tiles
[b] @ 700 MHz
[c] GOPS
[d] GOPS/W
[e] CUDA cores

3. METHODS, ASSUMPTIONS, AND PROCEDURES

In this section we describe our initial methods, assumptions, and procedures and discuss revisions to them that took place during the course of this research. We begin from the basic premise that hardware and software techniques for providing high performance and low energy consumption will be necessary to meet the growing demands of high performance energy efficient embedded computing (HPEEC). Hardware techniques have received significant attention from the hardware vendors, as well as in the literature. Therefore, our focus is on developing models for power and performance that can aid in the development of accurate, autonomous, and robust software techniques that will execute on HPEEC hardware .

3.1. Integrated Power and Performance Model

A significant motivation for this work is to enable processing as close to the sensing source as possible, particularly in contested and denied environments. While the idea of using cloud computing infrastructures to accommodate such processing has received significant attention recently, such techniques come with serious challenges in hostile environments [14]. Therefore, this work sought to develop new techniques that can assist with providing PED capabilities at the sensor using HPEEC technologies. In particular, an understanding of how certain applications will perform on specific HPEEC platforms is essential. These considerations must be made both in advance (i.e. global system design) and during runtime of the network (i.e. allow for dynamic adaptation of PED tasks). However, it should be clear that a meaningful and accurate technique to model power and performance for a variety of applications and architectures is needed to provide the basis of any analysis. Such a model has been proposed by Hong and Kim [15] for GPGPUs.

The primary assumption for the Integrated Power and Performance (IPP) model is that not every application will require all cores of a GPGPU to achieve maximum performance. In particular, the authors observed that certain types of applications will exhibit no further performance improvement by increasing the number of cores working on the task due to memory bandwidth limitations. The authors categorize GPGPU applications as either “bandwidth-limited” or “computationally intensive” which describe whether the peak performance is maximally-limited by the number of memory requests that can be concurrently handled, in the case of the former, or by the number of processing cores available, for the latter. In the case of

bandwidth-limited applications, the program uses some number of cores less than the maximum number of cores available when it reaches the bandwidth limitation. This is defined as the optimal number of cores (for computationally intensive programs, the optimal number of cores is the maximum number of cores). However, utilizing only the optimal number of cores leads to power inefficiency, since typically the inactive cores would still be powered.

The authors suggest that if the optimal number of cores can be accurately predicted in advance, this could allow for the excess cores to be powered off or otherwise disabled by hardware or a thread scheduler, and result in savings over the default case. Therefore, they propose a technique that integrates a power prediction model with application performance estimation, to enable power-efficient operation of various applications on a GPU.

However, the IPP model as it is proposed falls short of offering immediate tangible benefits for application developers. For one, the authors rely on some other technique (e.g. runtime thread scheduler) to actually utilize the results from IPP to achieve the power savings. Second, they do not investigate other ways in which processor usage can be optimized. Certainly the energy savings that can be gained by using IPP on the GPGPU improves energy efficiency, but it suggests a misuse of computing resources. That is, the application is not utilizing the full processor, and in a dynamic environment this can be costly if other tasks are unable to be handled. Extension of the IPP model proposed in [15] could allow for more efficient resource allocation, either by partitioning the GPGPU between multiple tasks or by identifying optimal processor types for handling specific applications.

Therefore, the basic underlying assumption of this project was that, while certain processing architectures perform very well on a wide-variety of applications, they may not necessarily be the optimal choice for specific applications. As such, we believe it is necessary to consider a heterogeneous mixture of architectures. To efficiently deploy and task such a heterogeneous mix, it would be necessary to characterize the performance and power behaviors of each processor given specific application requirements.

Thus, this project sought to extend the techniques devised in [15] for an emerging architecture, named Epiphany. The Epiphany Multicore Architecture is a multicore processor architecture developed by Adapteva, Inc. with design goals of high floating point performance and energy efficiency in mind. Furthermore, the Epiphany IP core design is highly scalable with the possibility of supporting from 16 to 4096 cores on a single chip. As shown in Table 1, the

Epiphany has demonstrated energy-efficient performance of 70.6 GFLOPS/Watt, which far exceeds the capability of other HPEEC candidates [10].

With an increasing emphasis in the DoD, in general, and the USAF, in particular, on the use of smaller, mobile, and even unmanned systems for sensing, communicating, and processing in the battlefield, these performance numbers make the Epiphany an intriguing candidate for potential applications in future agile, high performance systems. A thorough investigation is therefore necessary, to include measurement of actual performance of the architecture using a relevant application, as well as an understanding of the software development effort required to port applications to the Epiphany.

Shown in Figure 1, the Epiphany-III Multicore Evaluation Kit (EMEK3) consists of a 16-core microprocessor daughter card (Epiphany) and an Altera Stratix-III FPGA controller. The EMEK3 connects to a host computer running Ubuntu Linux via USB interface. We began our investigations for this project utilizing EMEK3; however as will be discussed in the following section this investigation was ultimately unsuccessful due in no small part to the perils of working with emerging, nonmature technology.

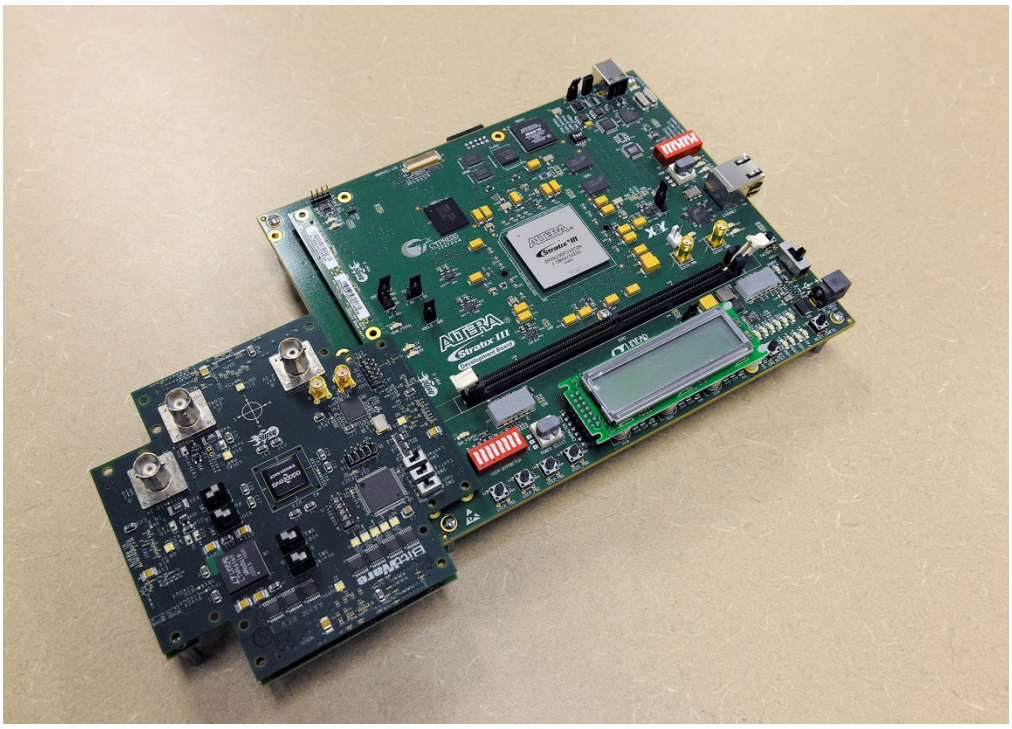
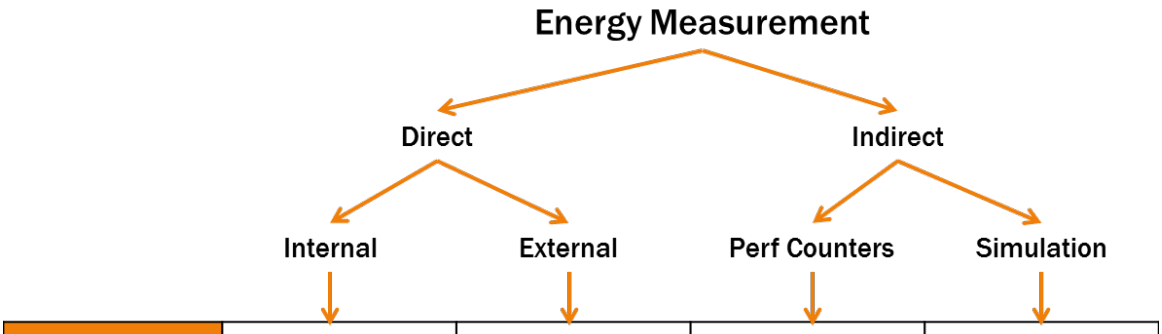


Figure 1 - Epiphany-III Multicore Evaluation Kit

3.2. Direct Hardware Measurement Method

During the initial phases of the project, it was determind that the techniques developed in the IPP model [15] would not be directly applicable for the EMEK3 architecture. For one, the EMEK3 provides very primitive support for software development, so the ability to analyze instruction use is limited to non-existent. In addition, the EMEK3 is configured with the Epiphany processor as a daughter card to an Altera FPGA. This contributes additional overhead that is not directly indicative of the Epiphany performance. Attempting to model this overhead would be mostly impractical effort, since an actual deployment of the Epiphany processor would not be configured in this way.

However, many other techniques for measuring and modeling power consumption have been developed and proven to be highly and sufficiently accurate. Some examples and taxonomy of these approaches are given in Figure 2. This chart illustrates a broad range of techniques, which is largely due to the inherent capabilities and limitations of the target hardware to be modeled. As an example, one of the more promising recent techniques is given in [16], which uses a back propagation artificial neural network (BPANN) training approach by indirectly measuring hardware performance event counters and on-chip sensors. The technique was found to demonstrate a high level of accuracy for predicting power consumption of NVIDIA C2075 GPUs by analyzing the relationship between certain hardware events and the GPU power consumption. An advantage of this technique over other approaches is that the model can be efficiently retrained for different architectures, assuming that similar performance counters are available.



Strength	Usability (simple)	Accuracy	Simple Ubiquitous Insights Scalable	Detailed insights Useful for architecture design
Weakness	Not widely available Few insights	Cumbersome Expensive Scalability issues	Require a model of power Sampling overhead	Cumbersome Ignore OS and I/O Can be slow Scalability issues
Examples	NVIDIA C2075 on-chip sensors	Data acquisition setup	Multiple Linear Regression Artificial Neural Network	Integrated Power and Performance

Figure 2 - Taxonomy of Approaches to Energy Measurement and Modeling [16]

Yet, even this technique is limited to certain types of hardware, especially emerging hardware for which low-level support functionality may not be fully matured. In particular, the Epiphany architecture does not provide profiling support to enable tracking of hardware event counters, nor does it offer an on-chip sensor for tracking system properties such as power, temperature, or memory use. Therefore, while the BPANN approach demonstrated among the highest accuracy in power and performance prediction and provides a mechanism that could be deployed in a runtime system, it was necessary to consider alternative techniques that might closely approximate this approach but be suitable for our hardware.

3.3. Hybrid Method

Realizing that existing approaches would not work with our particular capabilities, we decided to consider a hybrid approach in which we would combine observations from different architectures in order to develop a generic model of application power and performance behavior. Specifically, we sought to develop a portable implementation of our applications on a processor capable of low-level monitoring (e.g. C2075 GPU). Through the use of hardware

performance counters, we would develop an analytical profile of the macrocharacteristics of the application, such as global memory usage, local memory usage, total instructions executed, and so on. These characteristics would also be correlated to the power measured through the on-chip sensor.

We validate our observations through direct measurement of power and performance on the Epiphany processor. For this effort, we use the *Watts Up?* Pro ES (pictured in Figure 3), which is capable of measuring and logging power consumption at 1 Hz intervals.

Figure 3 - Watts Up? Pro ES

3.4. Wigner-Ville Distribution Implementation

As a demonstration of the applicability of the Epiphany processor architecture to the SIGINT application domain, it was necessary to implement SIGINT codes as part of this research effort. In collaboration with AFRL/RIGC engineers, we settled on using the Wigner-Ville Distribution (WVD) for time-frequency analysis of LPI radar signals, a relevant USAF application that is suitable for the HPEEC domain.

Detection of LPI signals is an important countermeasure technique, and WVD has been found to be particularly useful for analysis of LPI radar waveforms [17]. This is because the technique is capable of simultaneously representing both the time and frequency characteristics of a signal. Furthermore, WVD allows a signal analyst to extract the parameters of the LPI radar, which enables methods of counteracting or defeating the LPI radar.

$$W_x(t,\omega) = \int_{-\infty}^{\infty} x\left(t+\frac{\tau}{2}\right) x^*\left(t-\frac{\tau}{2}\right) e^{-i\omega\tau} d\tau \qquad (1)$$

Equation 1 gives the basic form of the WVD, where *x(t)* is the input signal and ω is the angular frequency, $2\pi f$ [17]. The implementation described here approximates this algorithm.

3.3.1 Matlab

The initial implementation being used for analysis and experimentation by AFRL/RIGC engineers was provided in Matlab code format. In addition, the implementation relies heavily on the Time-Frequency Toolbox and other built-in functionalities and libraries of Matlab. Such an implementation would not be suitable for HPEEC system application, particularly for the types of architectures listed in Table 1, due to both performance considerations and software licensing restrictions.

3.3.2 Sequential C

The first step of our development of a WVD implementation is to port the Matlab code to a sequential C implementation. Our assumption is that the sequential WVD implementation will be functionally equivalent to the Matlab code; however we may actually observe a decrease in performance. This is due to the fact that certain signal processing functions (e.g. fast Fourier Transform) have been heavily optimized in Matlab, while we exploit the open-source library FFTW3 [18]. However, the sequential implementation is a necessary first step towards developing the cross-platform, parallelized version of WVD that we discuss in the following section. We validated the functionality of the C implementation of WVD by comparing its output for several different signals with the output from the original Matlab code.

3.3.3 STDCL

The Epiphany Software Development Kit (SDK) supports OpenCL development, and this was exploited during the porting of the WVD application. This will enable future projects to benefit from significantly reduced development effort, while also providing a relative comparison of the strengths and weaknesses of each architecture for a particular application. However, the importance of the required development effort cannot be understated as it can be a significant inhibiting factor in large-scale deployment of HPEEC architectures, particularly when non-portable programming application programming interfaces (APIs) are utilized.

As mentioned previously, the utilization of the EMEK3 architecture had been a limiting factor in development of the WVD codes. The native SDK lacked support for efficient application development. Brown Deer Technology has developed the CO-Processing THReads (COPRTHR) SDK to support STandarD Compute Layer (STDCL) on the Epiphany processor [19]. STDCL is a simplified API that leverages OpenCL. We have utilized COPRTHR for this project because it builds upon OpenCL functionality to encapsulate and simplify many of the device-specific API calls and thus has a shorter learning curve than the native SDK. In addition, we gain portability by using the OpenCL-based COPRTHR SDK, mitigating some concerns mentioned in the preceding paragraph.

However, even when utilizing COPRTHR the EMEK3 was found to be very buggy and slow. Thus software development eventually stalled for this platform. Instead, we focused on developing the STDCL implementation of WVD on other processors (i.e. GPGPU), with the expectation that we could port the codes to other architectures in the future. We will discuss this further in Section 4.1. Similar to the C implementation, we validated the functionality of the STCDL implementation of WVD by comparing its output for several different signals with the output from the original Matlab code.

3.5. Speckle Reducing Anisotropic Diffusion Implementation

As discussed in Section 3.2, we adopted a hybrid approach to characterizing processor power and performance behavior. One component of this approach requires the ability to execute the same application on multiple architectures and correlate behavior characteristics on one architecture to a generic model which could be verified through execution on our target architecture (i.e., Epiphany).

The Rodinia benchmark suite [20][21] is a well-known and widely-used suite of kernels representing multiple relevant application domains for the analysis of heterogeneous processing architecture performance. While the effort involved to port all of the Rodinia benchmarks to STDCL is well beyond the scope of this project, we focused specifically on the speckle reducing anisotropic diffusion (SRAD) kernel because it falls within the Image Processing domain, and therefore is most suitable for the specific USAF application domain being studied here.

The SRAD implementation was ported by Dr. David Richie, Brown Deer Technology, under collaboration established through the DoD High Performance Computing Modernization Program (HPCMP) User Productivity Enhancement, Technology, and Training (PETTT) program. As the developer of COPRTHR and STDCL, Dr. Richie possesses the unique expertise to port the SRAD kernel to STDCL. Dr. Richie also has worked extensively with the Epiphany architecture and has the necessary knowledge to ensure optimal performance of the SRAD kernel.

4. RESULTS AND DISCUSSION

In this section we discuss the results of our research effort and provide analysis of observations made. We provide the basic model derived from our analysis and form a baseline from which future work can proceed, leveraging these results.

4.1. Parallella Processor

This project began with the intention of utilizing the Epiphany Multicore Evaluation Kit (EMEK3) for developing and evaluating the power and performance characteristics of the Epiphany IP core design. For reasons mentioned previously, in addition to the development of more advanced and mature products utilizing the core design, the effort shifted focus to the Parallella processor. AFRL/RITB procured a 16-core Parallella processor, similar to that shown in Figure 4, in June 2014. Under a no-cost extension to this project, and with the help of a High Performance Computing Internship Program (HIP) intern for the summer 2014, we modified the project objectives to examine the performance of this processor.

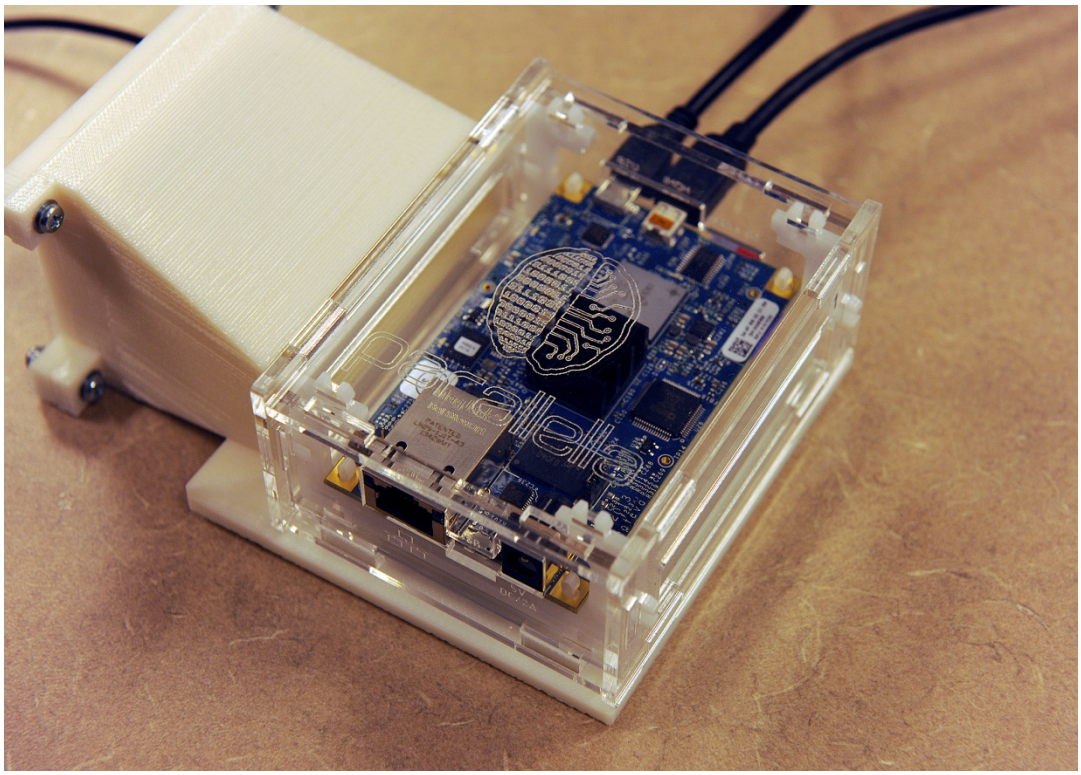


Figure 4 - Parallella processor

While the newer Parallella processor would provide us with more accurate assessment of power consumption and application behavior, its early study was fraught with difficulty and delays. In particular, the first board received ended up suffering an unrecoverable failure, for which we were unable to ascertain the cause. Fortunately, we were able to leverage multiple boards from another in-house research project. After troubleshooting some significant operational issues – we determined that the SD cards shipped with the processors containing the operating system were for a different hardware version – we were able to boot the Parallella and update the COPRTHR SDK to begin application testing.

Through the COPRTHR SDK, the STDCL API allows for significant portability advantages, not offered by other programming techniques. Thus by developing a single version of our applications, we can leverage multiple processing architectures and analyze the power and performance characteristics of each. What distinguishes COPRTHR and STDCL from other APIs, i.e. OpenCL, is specifically the ability to program for Epiphany-based devices. This compatibility was specifically developed by Brown Deer Technology to provide finer control

and precision on the Epiphany processor. Furthermore, because STDCL leverages OpenCL, we are able to develop and port codes written in STDCL to other processors (e.g. NVIDIA GPUs, Intel Xeon Phi, etc.) that support OpenCL-compatibility.

4.2. WVD Performance Evaluation

Due to schedule constraints caused by multiple delays in the project, a complete power and performance evaluation of the WVD code was not completed. Rather, only a performance evaluation was completed, and we present here the comparison of execution performance results between the C (sequential) and STDCL (parallel) implementations.

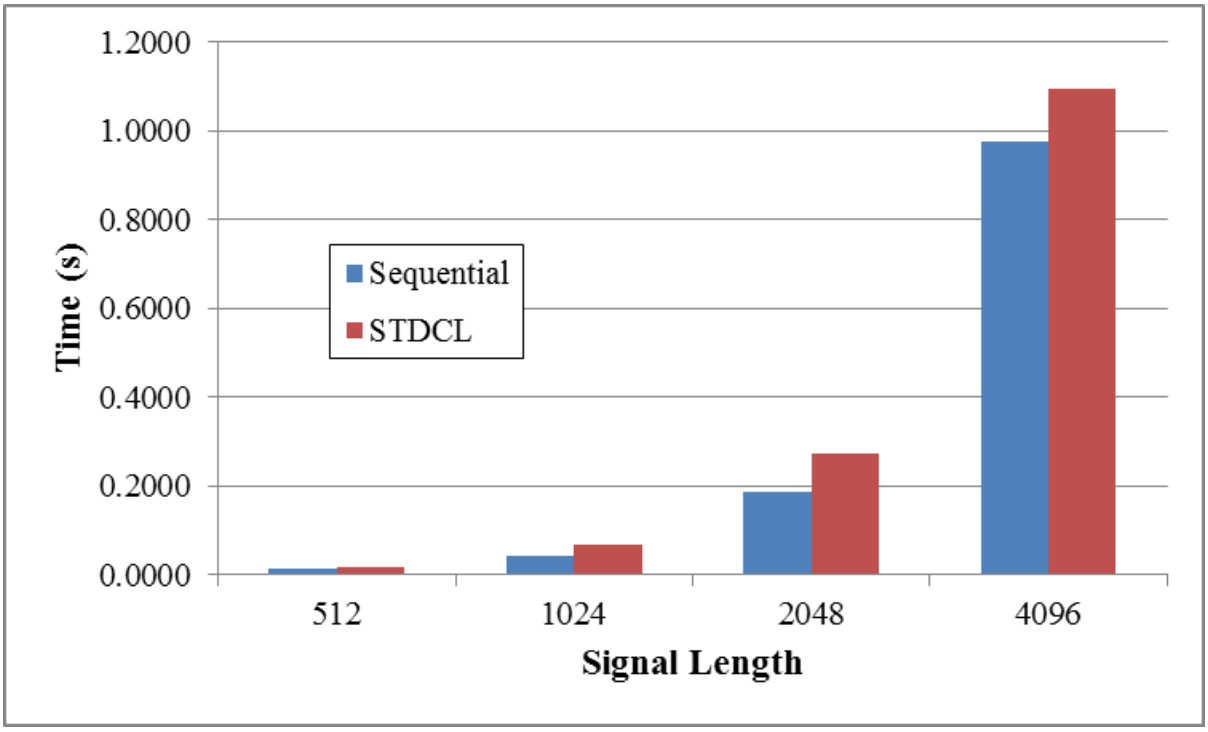


Figure 5 - Performance Comparison of C and STDCL implementations on GPGPU

These results show that in its current instantiation, the STDCL performs equivalently or worse than the sequential C implementation. We expect in general that a parallel STDCL code should outperform a sequential implementation, but note that neither implementation has undergone significant optimization. In particular for the parallel implementation, there are multiple factors that contribute to the performance degradation. The most significant is that the

time for memory transfers between the host (CPU) and device (GPU) increases linearly as the signal length gets larger. We observe that the actual kernel computation time accounts for a small fraction of the overall execution time. Figure 7 shows the decomposition of total execution time between kernel execution time and memory transfer time. The figure shows that although kernel execution time increases linearly with signal length, the total application execution time is quickly dominated by memory transfer times and other overhead, as the kernel accounts for less than 1% of the total execution time for signal length of 4096.

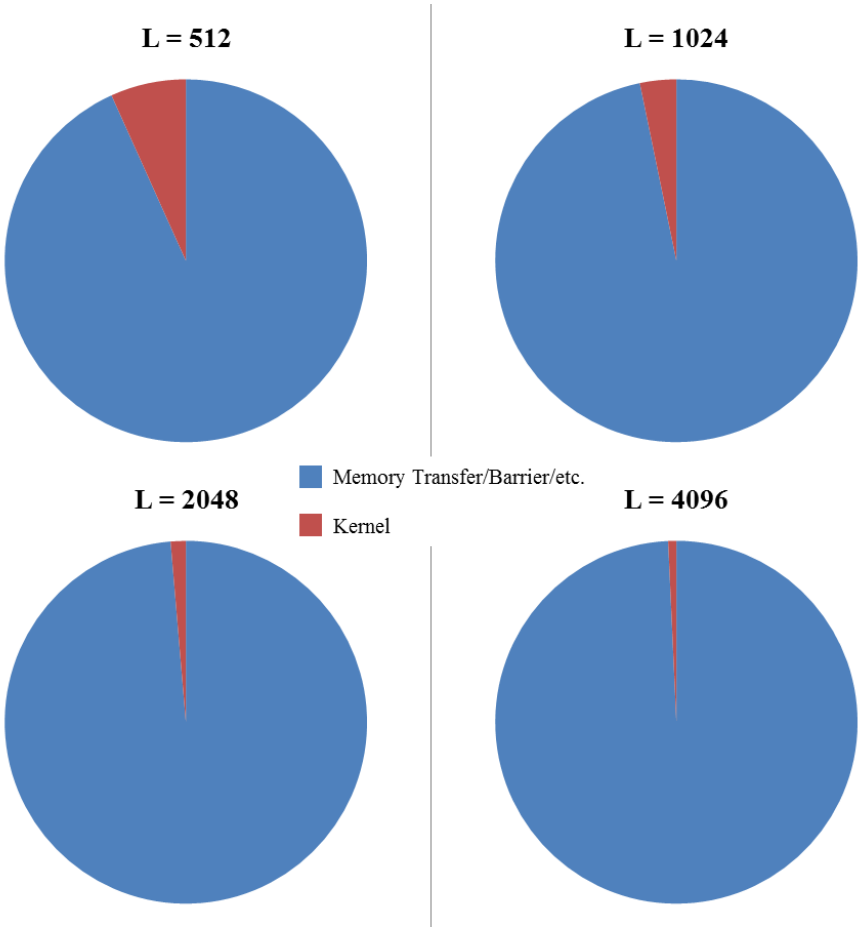


Figure 6 - Decomposition of Total Execution Time for Parallel WVD on Signals of Various Lengths, *L*

In contrast, the total execution time of the sequential WVD implementation is all computation time. Therefore, in terms of pure WVD computation the parallel implementation gives a performance speedup of at least 10X, as shown in Table 2.

Table 2 - Parallel Speedup of WVD Computation

Signal Length	Sequential	Parallel (kernel only)	Speedup
512	.0124	.0012	10.3
1024	.0421	.0021	19.9
2048	.1885	.0037	50.7
4096	.9756	.0084	116.7

Future research will address optimizations of the WVD kernel, to include better memory layout and use to limit the costly transfer times. In addition, a significant part of the WVD application is a Fast Fourier Transform (FFT) and the WVD kernel computation. For both sequential and parallel implementations we utilize the FFTW3 library [18], but expect that additional performance gains can be made by implementing a parallel FFT. More details will be provided in Section 5.

4.3. SRAD Power and Performance Evaluation

Using the STDCL implementation discussed above in Section 3.4, we analyze the performance and power behavior of the SRAD kernel on Parallella by executing several iterations of the kernel and logging the power consumption using a *WattsUp?* meter. The code developed for this effort has inline instrumentation to measure the kernel loop time, as well as memory transfer times (i.e., copy between host and device memory), which we use for the execution performance results. The overhead for this instrumentation is negligible with respect to the execution performance of the application.

We observed that the Parallella power consumption varies minimally during the execution of the SRAD application. The baseline idle Parallella power consumption is measured to be 6.3 W. During execution of the kernel, the power varies between 6.3 and 6.4 W. While this is good from the perspective of energy efficiency, it presents a challenge for modeling the kernel power consumption. In order to determine if this behavior is typical, we will need to execute additional kernels on Parallella, as well as execute the SRAD STDCL implementation on

different HPEEC architectures. Neither of these ideas were studied under this project, but would be candidate topics for future research.

The program was also modified to accept varying block and thread configurations for the kernel execution. The default case of 16 work-items (i.e. threads) in one work group (i.e. block) is determined to be the optimal configuration on Parallella. However, the variations made allow for insight into the sensitivity of the kernel power and performance to optimal block configuration for a specific processing architecture. This behavior is under further study as follow-on to the effort being reported here, but we provide preliminary observations from the SRAD kernel on Parallella.

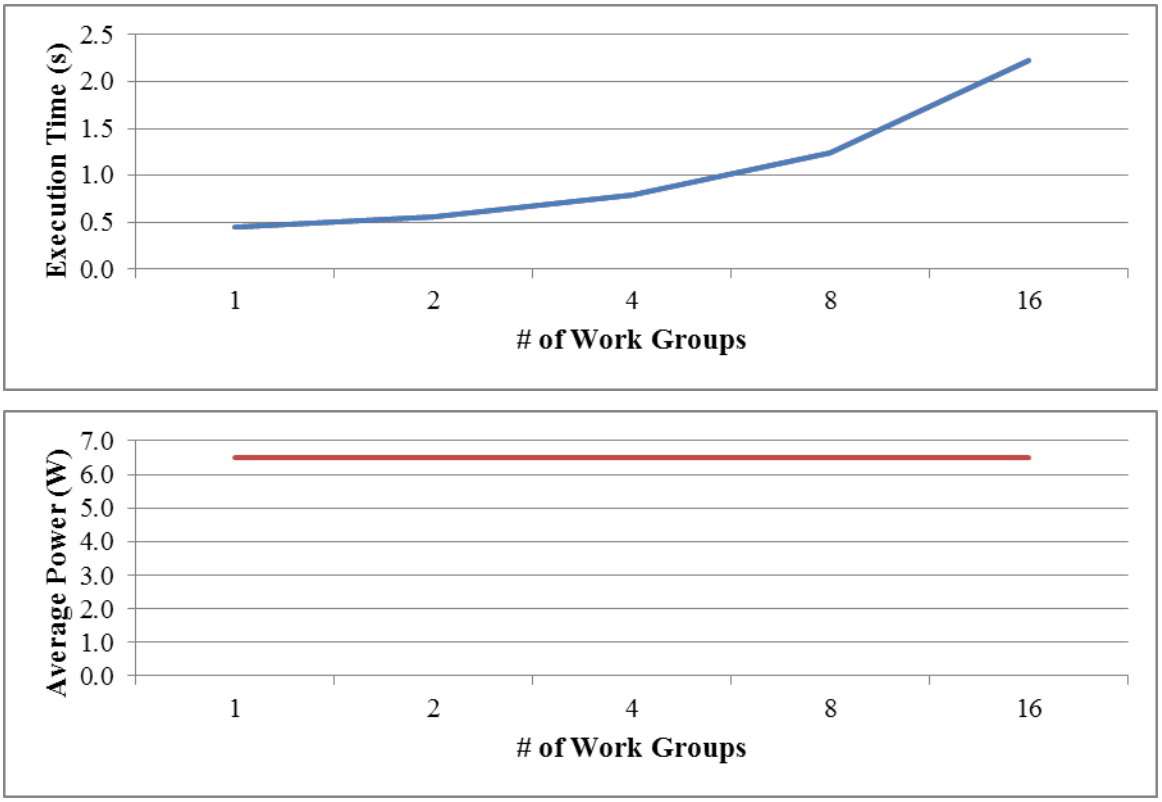


Figure 7 - SRAD Power and Performance Comparison with Varying Number of Work Groups

As can be seen in Figure 7, the execution performance of the application is affected by the configuration of the kernel range, while the power consumption remains constant across all instances. This suggests that in terms of performance portability of codes we must consider the

impact of work group dimensions with respect to the specific architectures to enable fair and reasonable comparison.

4.4. GPU Modeling and Analysis

With the advantage of portability, as discussed in the previous section, and due to EMEK difficulties and Parallella procurement delays, we proceeded with development of our application codes on GPU-based workstations. While, we are certain that application portability does not imply performance portability due to inherent differences in the architecture designs, the development effort was beneficial to provide a greater understanding of, and experience using, STDCL.

5. CONCLUSIONS

The effort reported here did not achieve its original stated goals, largely due to delays in development on the EMEK3 platform, and subsequently with delivery and initial testing on the Parallella board. These delays were explained in Section 4.1, and led to not being able to begin evaluation of our target applications on the Parallella until the final months of the project. This limited our ability to use the experimental results to develop power and performance models.

However, the effort was invaluable in the process of learning to develop and/or port application codes to the STDCL domain. This will benefit future research in the area of HPEEC architectures because STDCL provides for greater portability. As such, we expect to continue the work started here to develop more kernels in STDCL and examine power and performance characteristics of multiple HPEEC architectures. In addition to the Parallella board, we will continue to examine performance on NVIDIA GPUs. We would also like to experiment with the NVIDIA Jetson TK1 (see Figure 8), which consists of a NVIDIA Tegra K1 System-on-Chip (SoC) which includes an ARM Cortex A15 and a Kepler GPU with 192 “CUDA” cores, and is more comparable as an HPEEC platform than Tesla series GPUs, such as the NVIDIA C2075 and K20. Jetson is particularly interesting because it features a unified memory architecture that eliminates data transfer overhead between the host and the GPGPU that was identified as the primary performance limitation. However, currently the Jetson SDK does not support OpenCL, so it would not be able to support STDCL. Some effort towards enabling STDCL through CUDA may be explored, as code portability remains a critical concern.

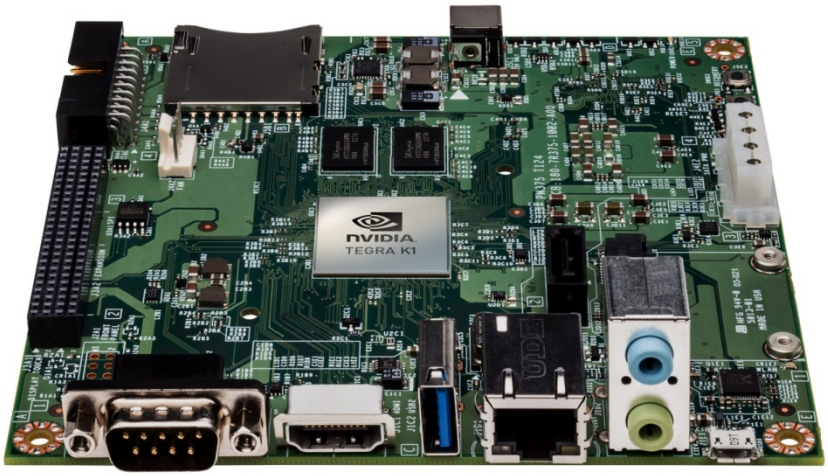


Figure 8 - NVIDIA Jetson TK1 Platform [22]

We also have remaining effort to do with respect to optimization of the codes that were ported to STDCL, particularly with the WVD application. As mentioned above, we have not tried to port the FFT to STDCL, but expect that this would provide even more performance gains over the sequential implementation.

Finally, we proposed two different methodologies for developing accurate power and performance models but found that, with the target HPEEC architectures studied here, they would be very challenging or impossible to apply. For example, applying the technique described in [16] would not be possible for Parallella because of the lack of profiler support to provide hardware event activity counts. In addition, if the observed kernel execution power consumption shows negligible variation from the baseline idle consumption, it may be difficult to correlate application activities to power consumption. We will explore new, system-agnostic techniques for analyzing power and performance across disparate architectures.

While this project did not yield the models and results that we had hoped, the experimentation and experience gained will greatly benefit research into the HPEEC area, specifically as they apply to SIGINT applications. Our hope is that future research driven by the results we did get will benefit significantly.

6. REFERENCES

[1] S. Magnuson, “Military ‘Swimming in Sensors and Drowning in Data’,” National Defense, January 2010.

[2] J. McLean and H. Schue, “Sensor Data Exploitation”, Jun. 17, 2011.

[3] Joint Publication 3-0, “Joint Operations”, Aug. 11, 2011. http://www.dtic.mil/doctrine/new_pubs/jp3_0.pdf

[4] Lt Col L.D. Hill, “An Airman’s View of NTISR,” Air Land Sea Bulletin, Issue 2007-3, September 2007. Retrieved from http://www.alsa.mil/library/alsb/ALSB%202007-3.pdf

[5] “Signals Intelligence,” National Security Agency/Central Security Service, Copyright 2009 National Security Agency. Accessed Jan. 23, 2013. http://www.nsa.gov/sigint/index.shtml

[6] A. Munir, S. Ranka, and A. Gordon-Ross, “High-Performance Energy-Efficient Multi-core Embedded Computing,” IEEE Trans. Parallel Distrib. Syst., vol.23, no.4, pp.684-700, April 2012

[7] NVIDIA Tesla K-Series Datasheet, October 2012.

[8] NVIDIA Tesla K20 GPU Accelerator Board Specification, November 2012.

[9] A. Shah, “Intel ships 60-core Xeon Phi processor,” Computerworld, Nov. 12, 2012. Accessed Jan. 23, 2013. http://www.computerworld.com/s/article/9233498/Intel_ships_60_core_Xeon_Phi_processor

[10] L. Gwennap, “Adapteva: More FLOPS, Less Watts: Epiphany Offers Floating-Point Accelerator for Mobile Processors,” Microprocessor Report, June 2011. Retrieved from: http://www.linleygroup.com/newsletters/newsletter_detail.php?num=4716

[11] T. Wimberly, “Tegra 3 saves more power than it consumes, DIDIM technology reduces backlight power by 40%,” Android and Me, Nov. 18, 2011. Accessed Jan. 23, 2013. http://androidandme.com/2011/11/news/tegra-3-saves-more-power-than-it-consumes-didim-technology-reduces-backlight-power-by-40/

[12] “Tegra 3 Multi-Core Super-Chip Processors,” NVIDIA, Copyright 2013 NVIDIA Corporation. Accessed Jan. 23, 2013. http://www.nvidia.com/object/tegra-3-processor.html

[13] “NVIDIA Quadro 1000M,” Notebook Check. Accessed Jan. 23, 2013. http://www.notebookcheck.net/NVIDIA-Quadro-1000M.47317.0.html

[14] K. Ha, G. Lewis, S. Simanta, and M. Satyanarayanan, “Cloud Offload in Hostile Environments,” School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, CMU-CS-11-146, Dec. 2011

[15] S. Hong and H. Kim, “An Integrated GPU Power and Performance Model,” Proceedings of the 37th Annual International Symposium on Computer Architecture, pp. 280-289, 2010

[16] S. Song, C. Su, B. Rountree, K.W. Cameron, "A Simplified and Accurate Model of Power-Performance Efficiency on Emergent GPU Architectures," Parallel & Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on , vol., no., pp.673,686, 20-24 May 2013.

[17] P. E. Pace, Detecting and Classifying Low Probability of Intercept Radars, Artech House, 2009.

[18] Frigo, M.; Johnson, S.G., "The Design and Implementation of FFTW3," *Proceedings of the IEEE* , vol.93, no.2, pp.216,231, Feb. 2005

[19] Brown Deer Technology, “COPRTHR SDK,” http://www.browndeertechnology.com/coprthr.htm

[20] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee, and K. Skadron. Rodinia: A Benchmark Suite for Heterogeneous Computing. In Proceedings of the IEEE International Symposium on Workload Characterization (IISWC), pp. 44-54, Oct. 2009.

[21] S. Che, J. W. Sheaffer, M. Boyer, L. G. Szafaryn, L. Wang, and K. Skadron. A Characterization of the Rodinia Benchmark Suite with Comparison to Contemporary CMP Workloads. In Proceedings of the IEEE International Symposium on Workload Characterization, Dec. 2010.

[22] R. Smith “NVIDIA Announces Jetson TK1 Dev Board; Adds Erista to Tegra Roadmap,” AnandTech, March 27, 2014. http://www.anandtech.com/show/7905/nvidia-announces-jetson-tk1-dev-board-adds-erista-to-tegra-roadmap

LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS

API	application programming interface
BPANN	back propagation artificial neural network
CARMA	CUDA on ARM Architecture
COPRTHR	CO-PRocessing THReads
DoD	Department of Defense
EMEK3	Epiphany-III Multicore Evaluation Kit
FFT	fast Fourier Transform
FFTW3	Fastest Fourier Transform in the West, version 3
FLOP	floating point operation
FMV	full motion video
FPGA	field-programmable gate array
GPGPU	graphics processing unit
HIP	High Performance Computing Internship Program
HPCMP	High Performance Computing Modernization Program
HPEC	high performance embedded computing
HPEEC	high performance energy-efficient embedded computing
IP	intellectual property
IPP	Integrated Power and Performance
ISR	intelligence, surveillance, and reconnaissance
LPD	low probability of detection
LPI	low probability of intercept
NTISR	non-traditional intelligence, surveillance, and reconnaissance
PCPAD	planning and direction, collection, processing and exploitation, analysis and production, dissemination
PED	processing, exploitation, and dissemination
PETTT	User Productivity Enhancement, Technology Transfer and Training
SAB	Scientific Advisory Board
SDK	software development kit
SIGINT	signals intelligence
SRAD	speckle reducing anisotropic diffusion
STDCL	STandarD Compute Layer
SWAP	size, weight, and power
USAF	United States Air Force
WAMI	wide area motion imagery
WVD	Wigner-Ville Distribution